Algorithmic Aspects of Heterogeneous Biological Networks Comparison

Guillaume Blin, Guillaume Fertin, Hafedh Mohamed-Babou, Irena Rusu, Florian Sikora, Stéphane Vialette

Algorithmic Aspects of Heterogeneous Biological Networks Comparison *

Guillaume Blin[1], Guillaume Fertin[2], Hafedh Mohamed-Babou[2], Irena Rusu[2], Florian Sikora[1], Stéphane Vialette[1]

[1] Université Paris-Est, LIGM - UMR CNRS 8049, France.
{Guillaume.Blin,Florian.Sikora,Stephane.Vialette}@univ-mlv.fr
[2] Université de Nantes, LINA - UMR CNRS 6241, France.
{Guillaume.Fertin,Hafedh.Mohamed-Babou,Irena.Rusu}@univ-nantes.fr

Abstract. Biological networks are commonly used to model molecular activity within the cell. Recent experimental studies have shown that the detection of conserved subnetworks across several networks, coming from different organisms, may allow the discovery of disease pathways and prediction of protein functions. There already exist automatic methods that allow to search for conserved subnetworks using networks alignment; unfortunately, these methods are limited to networks of same type, thus having the same graph representation. Towards overcoming this limitation, a unified framework for pairwise comparison and analysis of networks with different graph representations (in particular, a directed acyclic graph D and an undirected graph G over the same set of vertices) is presented in [4]. We consider here a related problem called k-DAGCC: given a directed graph D and an undirected graph G on the same set V of vertices, and an integer k, does there exist sets of vertices $V_1, V_2, \ldots V_{k'}$, $k' \leq k$ such that, for each $1 \leq i \leq k'$, (i) $D[V_i]$ is a DAG and (ii) $G[V_i]$ is connected ? Two variants of k-DAGCC are of interest: (a) the V_is must form a *partition* of V, or (b) the V_is must form a *cover* of V. We study the computational complexity of both variants of k-DAGCC and, depending on the constraints imposed on the input, provide several polynomial-time algorithms, hardness and inapproximability results.

1 Introduction

Recent advances in the field of biological networks comparison, inspired by developments in sequence comparison, have provided a prominent tool for explaining organization and interpreting function and evolution of biological networks [20]. In their most basic abstraction level, biological networks can be modeled by graphs, where vertices represent cellular compounds, and edges (or arcs) represent their interactions. These graphs may be directed or undirected, depending on the type of networks they represent: for instance, a Protein-Protein Interaction network (PPI) is usually modeled by an undirected graph whose vertices

* This work was partially supported by GDR-IM and ANR project BIRDS JCJC SIMI 2-2010

are proteins and edges represent the physical interactions between proteins; however, a metabolic network is modeled by a directed graph (also called *reaction graph*), where (a) any vertex represents a reaction which needs both a set of input compounds (substrates) and some enzymes (proteins) as catalysts, and produces a set of compounds (products) and (b) there is an arc from reaction r_i to reaction r_j if r_j uses, as substrate, a product of r_i.

Each biological network represents a partial view of the molecular activity within the cell. Thus, comparing them provides a better understanding of the behavior of a biologic system. We say that two biological networks are *homogeneous* if they are of the same type and have the same graph representation, but come from different species. In contrast, networks are said to be *heterogeneous* if they come from the same species, but are of different types and have different graph representations (e.g., a directed graph and an undirected graph).

The main comparative approaches considered so far focus on homogeneous networks using *networks alignment* (see, among others, [10,11,18,20,5,2,21,12]). Such methods are powerful for detecting conserved modules across several networks of different species. Unlike homogeneous networks, only few research works aim at comparing heterogeneous networks. Most of the existing methods used to compare heterogeneous networks are usually manual, or use case-by-case methods. In [22,19,13,16,1], authors search for a chain of reactions in a metabolic network, by investigating the relationships between genes involved in the reactions and the proximity of genes along the genome. Interactions between proteins that catalyze successive reactions in metabolic networks have been studied in [3,9]. In [14,8,6], authors search signaling pathways by confronting a signal transduction network, represented by a directed graph, to a PPI network, represented by an undirected graph. They perform an orientation of the PPI graph as follows: given a list of ordered source-target pairs, the PPI graph is oriented in such a way that, for a maximum number of pairs (s, t), there is a directed path from the source s to the target t. The advantage of this method is that it allows to return to homogeneous networks with the same graph. However, in addition to its computational difficulty, the good behavior of this method is closely related to the choice of the list of the source-target pairs. In [8], such a list is manually constructed.

Recently, a unified framework for comparison and analysis of heterogeneous networks was proposed [4]. In particular, the central problem considered in [4] is the following: given a directed acyclic graph (DAG) D' and an undirected graph G' built on the same set of vertices, find the longest directed path in D' whose vertices induce a connected component in G'. In this setting, D' and G' represent a metabolic network and a PPI network, respectively. However, in general, metabolic networks are not DAGs, and hence contain directed cycles.

Therefore, we consider in thispaper the following problem, called k-DAGCC, that can be seen as a way to pre-process the input graphs in [4]: given a directed graph D and an undirected graph G on the same set V of vertices and an integer k, does there exist sets of vertices $V_1 \dots V_{k'}$, $k' \leq k$, such that, for each $1 \leq i \leq k'$, (i) $D[V_i]$ is a DAG and (ii) $G[V_i]$ is connected ? Two variants of k-DAGCC are

considered: (a) the *partition* version, where the V_is must form a partition of V, and (b) the *cover* version, where the V_is must form a cover of V.

The problem k-DAGCC (under both its variants) is also motivated by the research of pathways in metabolic networks that correspond to functionally related proteins in PPI networks. Indeed, pathways in metabolic networks correspond to DAGs [18], while functionally related proteins in PPI networks correspond to connected subgraphs [1,2,15,21,12]). Thus, it is of interest to be able to extract biologically relevant information from two large networks by decomposing them into smaller modules that each (i) carry a rich biological information and (ii) are easier to interpret.

In this paper, we study the computational complexity of both variants of k-DAGCC and, depending on the constraints imposed on the input, provide several polynomial-time algorithms, along with hardness and inapproximability results. Once the main notations and definitions will be stated (Section 2), we will define formally and study in detail the complexity of the two variants of the k-DAGCC problem, first in its partition version (Section 3), then in its cover version (Section 4).

2 Preliminaries

This paper is concerned with both directed and undirected graphs. We briefly recall the basic needed material.

A *graph* $G = (V, E)$ consists of a set of vertices V and a set of edges (unordered pairs of vertices) E. To shorten the exposition, we shall usually abbreviate $|V|$ and $|E|$ to n and m, respectively. The *degree* of a vertex $u \in V$ is the number of edges incident to it. A graph is *acyclic* if it does not contain any cycle. An *independent set* is a subset $V' \subseteq V$ such that $(x, y) \notin E$ for any $x, y \in V'$. A graph is *connected* if there exists a path between any pair of vertices. For any $V' \subseteq V$, we let $G[V'] = (V', E')$ stand for the *subgraph induced* by V', i.e. $E' = \{\{x, y\} \in E : x, y \in V'\}$. A graph is a tree if it is both connected and acyclic. A *star* of order $n \geq 3$ is a tree with exactly one vertex of degree strictly greater than 1. A graph is *complete* if it contains all possible edges. A *planar* graph is a graph that can be embedded onto the plane, in such a way that its edges intersect only at their endpoints. Finally, a graph is said to be *outerplanar* if it has a planar embedding such that the vertices lie on a circle and the edges lie inside that circle, without crossing each other.

A *directed graph* $D = (V, A)$ consists of a set of vertices V and a set of arcs (ordered pairs of vertices) A. A *directed acyclic graph* (DAG) is a directed graph that does not have any directed cycle. For any $V' \subseteq V$, we let $D[V'] = (V', A')$ stand for the *subgraph induced* by V', i.e. $A' = \{(x, y) \in A : x, y \in V'\}$.

We shall consider graphs and directed graphs defined on the same set of vertices, i.e. we will write $D = (V, A)$ for the directed graph and $G = (V, E)$ for the undirected graph. To avoid ambiguity, we let m_D and m_G stand for $|A|$ and $|E|$, respectively. Given two such graphs $D = (V, A)$ and $G = (V, E)$ built on the same set V of vertices, we say that a partition (resp. a cover) $\{V_1, V_2 \dots V_k\}$ of

V is *valid* if, for any $1 \leq i \leq k$, $D[V_i]$ is a DAG and $G[V_i]$ is connected. The two variants of k-DAGCC we are going to study in this paper are called respectively k-DAGCC-Partition and k-DAGCC-Cover, and are defined as follows.

k-DAGCC-Partition
Instance: A directed graph $D = (V, A)$, an undirected graph $G = (V, E)$, and an integer k.
Question: Does there exist a valid partition $\mathcal{P} = \{V_1, V_2, \ldots, V_{k'}\}$ of V such that $k' \leq k$?

k-DAGCC-Cover
Instance: A directed graph $D = (V, A)$, an undirected graph $G = (V, E)$, and an integer k.
Question: Does there exist a valid cover $\mathcal{C} = \{V_1, V_2, \ldots, V_{k'}\}$ of V such that $k' \leq k$?

The natural minimization version of the above decision problems are denoted Min-DAGCC-Partition and Min-DAGCC-Cover, respectively.

k \ G	Graph	Outerplanar	Tree	Star	Path
$(n-k) = \mathcal{O}(1)$	P [Prop. 2]				
$k = \mathcal{O}(1)$	NPC [Prop. 4]	P [Prop. 3]			
k unbounded	Inapprox. within $n^{1-\epsilon}$ [Prop. 6]	APX-hard [Prop. 7]		NPC [Prop. 5]	P [Prop. 1]

Table 1. Complexity results for k-DAGCC-Partition and Min-DAGCC-Partition.

k \ G	Graph	Outerplanar	Tree	Star	Path
$k = \mathcal{O}(1)$	NPC [Prop. 9]				P [Prop. 8]
k unbounded	Inapprox. within $n^{1-\epsilon}$ [Prop. 10]				P [Prop. 8]

Table 2. Complexity results for k-DAGCC-Cover and Min-DAGCC-Cover.

The results presented in this paper are summarized in Table 1 (for problems k-DAGCC-Partition and Min-DAGCC-Partition) and Table 2 (for problems k-DAGCC-Cover and Min-DAGCC-Cover).

3 Partition Version of k-DAGCC

In this section, we provide polynomial-time algorithms, hardness and inapproximability results for different variants of Min-DAGCC-Partition and k-DAGCC-Partition (see Table 1). We first provide polynomial-time algorithms for three restricted cases: (a) G is a path, (b) $n-k$ is a constant, and (c) G is an outerplanar graph and k is a constant.

Recall that, given a directed graph $D = (V, A)$ and an undirected graph $G = (V, E)$, testing whether D is a DAG (resp. whether G is connected) can be done in $O(n + m_D)$ time (resp. $O(n + m_G)$ time) by depth-first search.

Proposition 1. Min-DAGCC-Partition *is polynomial-time solvable when G is a path.*

Proof. The proof is by a simple greedy algorithm. Write $(v_1, v_2, \ldots, v_n)$ for the path G. We consider the vertices of G from v_1 to v_n. We start with $S = \emptyset$. For vertex v_i, if $D[S \cup \{v_i\}]$ is a DAG, we add v_i to S, otherwise we report S as an element of the sought partition and set $S = \{v_i\}$. When the algorithm stops, we report the current S as the last element of the partition.

It is easily seen that this greedy algorithm produces a valid partition of V. What is left is to prove that the valid partition obtained by the above algorithm is of minimum cardinality. Let $\mathcal{P} = \{V_1, V_2, \ldots V_k\}$ be the partition returned by our algorithm, and suppose, aiming at a contradiction, that there exists a strictly smaller valid partition $\mathcal{P}' = \{V'_1, V'_2, \ldots, V'_\ell\}$, thus with $\ell < k$. We first observe that, by construction, V_1 cannot be a proper subset of V'_1. Since any connected subgraph of G is built on consecutively indexed vertices $\{v_p, v_{p+1}, v_{p+2} \ldots v_q\}$, it follows that there must exist $2 \leq i < k$ and $2 \leq j \leq \ell$ such that (1) V_i is a proper subset of V'_j, and (2) V'_j contains the first vertex, say x, of V_{i+1}. This is a contradiction since $V_i \cup \{x\}$ is not a DAG in D. □

Proposition 2. k-DAGCC-Partition *is polynomial-time solvable when $n-k$ is a constant.*

Proof. Let us prove that this variant of k-DAGCC-Partition can be solved in polynomial time by an exhaustive procedure. We consider all k-partitions of V. We start from the unique n-partition of V and iteratively compute all $(k-1)$-partitions of V starting from its k-partitions. Recall that the Stirling number of the second kind $S(n, k)$ represents the number of ways to partition a set of n elements into k nonempty subsets. We have $S(n, k) = S(n, n) \prod_{i=0}^{n-k-1} \binom{n-i}{2}$. But $S(n, n) = 1$, and hence $S(n, k) = \prod_{i=0}^{n-k-1} \binom{n-i}{2} = O(n^{2(n-k)})$. For each of these $O(n^{2(n-k)})$ partitions, we can check in polynomial-time whether it is a valid solution for k-DAGCC-Partition. The proposition follows. □

Proposition 3. k-DAGCC-Partition *is polynomial-time solvable when G is an outerplanar graph and k is a constant.*

Proof. Recall that a graph is outerplanar just when its vertices can be placed on a circle so that its edges become non-crossing chords of the circle. Fix any such an outerplanar circular embedding. Let $V_1, V_2, \ldots, V_k$ be any valid partition of V. Now, associate to any V_i, $1 \leq i \leq k$, the smallest arc of the circle that covers all vertices in V_i (according to the outerplanar circular embedding). Since $V_1, V_2, \ldots, V_k$ is a valid partition of V, any vertex is covered by at least one arc of the circle, and no two arcs of the circle share a common endpoint. Moreover, since $(v_1, v_2, \ldots, v_n)$ is an outerplanar embedding of G and $G[V_i]$ is connected for every $1 \leq i \leq k$, it follows that no two arcs of the circle properly overlap.

The algorithm is by enumerating all sets of non-overlapping k arcs of the circle with distinct endpoints that cover V. Let $v_1, v_2, \ldots, v_n$ be the vertices of G in the outerplanar circular embedding following, say, the trigonometric orientation. For any $1 \leq i \leq n$, consider all $2(k-1)$-subsets of $V \setminus \{v_i, v_{i-1}\}$ (by convention, $v_0 = v_n$). Let $\{v_{j_1}, v_{j_2}, \ldots, v_{j_{2(k-1)}}\}$ be such a subset, where the vertices are read along the outerplanar circular embedding starting from v_i and following the trigonometric orientation. We now need to construct all possible arcs of the circle with endpoints $v_i, v_{j_1}, v_{j_2}, \ldots, v_{j_{2(k-1)}}, v_{i-1}$ such that no two arcs of the circle are overlapping. Clearly, this reduces to considering all well-formed parenthesis strings of length $2k$. Our construction yields sets of k arcs $a_1, a_2, \ldots, a_k$ of the circle such that (1) any two arcs of the circle have distinct endpoints, (2) no two arcs of the circle are overlapping, and (3) each vertex of V is covered by at least one arc of the circle. For any such solution, we construct a partition $(V_1, V_2, \ldots, V_k)$ as follows: V_i contains all vertices covered by arc a_i of the circle, except those vertices covered by at least one arc a_j of the circle which is strictly covered by a_i. This algorithm is $O(n^2 \binom{n}{2(k-1)} \binom{2(k-1)}{k-1} (n+m))$ time, where $m = \max\{m_D, m_G\}$. Indeed, we have n possibilities for choosing the reference vertex v_i. Starting from each reference vertex v_i, we consider $\binom{n}{2(k-1)} \frac{1}{k} \binom{2(k-1)}{k-1}$ sets of k arcs of the circle ($\frac{1}{k} \binom{2(k-1)}{k-1}$ is the number of well-formed parenthesis strings of length $2k$). From these sets, one can construct a solution V_i, $1 \leq i \leq k$, is $O(n)$ time, and check whether each subset V_i, $1 \leq i \leq k$, induces both a DAG in D and a connected component in G in $O(n+m)$ time. The proposition follows. □

We now prove that k-DAGCC-Partition becomes NP-complete when (a) G is a complete graph or (b) G is a star.

Proposition 4. *For any $k \geq 2$, k-DAGCC-Partition is NP-complete even when G is a complete graph.*

Proof. k-DAGCC-Partition is certainly in NP. To prove hardness, we propose a reduction from the NP-complete Not-All-Equal 3SAT (NAE 3Sat) problem [7]: Given a collection $\mathcal{C}_q = \{c_1, \ldots c_q\}$ of q clauses, where each clause consists of a set of three literals over a finite set of n boolean variables $\mathcal{V}_n = \{x_1, \ldots x_n\}$, is there a truth assignment of each variable of $\mathcal{V}_n$ such that at least one of the literals is `true` and at least one is `false` in each clause ? For ease of exposition, for any $1 \leq j \leq 3$, we let c_i^j stand for the j-th literal of clause c_i.

Given any instance $(\mathcal{C}_q, \mathcal{V}_n)$ of NAE 3Sat, we build D and G as follows:

- $V = \{v_i^j : 1 \leq i \leq q, 1 \leq j \leq 3\} \cup \{v_i : 3 \leq i \leq k\}$
- $A = \{(v_i^j, v_{i'}^{j'}), (v_{i'}^{j'}, v_i^j) : c_i^j = \overline{c_{i'}^{j'}}\} \cup \{(v_i^1, v_i^2), (v_i^2, v_i^3), (v_i^3, v_i^1) : 1 \leq i \leq q\} \cup \{(v_i, v), (v, v_i) : 3 \leq i \leq k, v \in V \setminus \{v_i\}\}$
- $E = \{(v, v') : v, v' \in V\}$

Roughly speaking, vertex v_i^j corresponds to the j-th literal of clause c_i, whereas vertices v_i, $3 \leq i \leq k$, are gadgets to adapt the proof for any $k \geq 2$. There is a cycle of length two between any pair of vertices representing two complementary literals (e.g. x_i and $\overline{x_i}$) and between v_i, $3 \leq i \leq k$, and any other vertex of V. Moreover, there is a cycle of length three between the triple of vertices (v_i^1, v_i^2, v_i^3) corresponding to the three literals of any clause c_i. Finally, G is a complete graph on V. An illustration of such a construction – omitting G – is given in Figure 1.

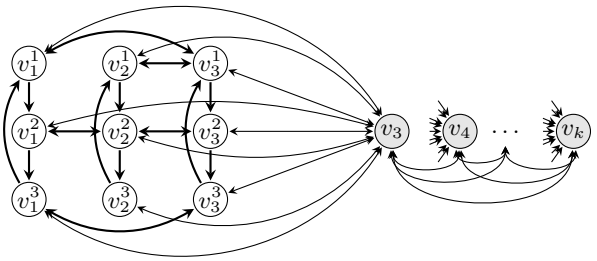

Fig. 1. Illustration of the construction of D, given $\mathcal{C}_q = \{(x_1 \vee x_2 \vee x_3), (x_1 \vee \overline{x_2} \vee x_4), (\overline{x_1} \vee x_2 \vee \overline{x_3})\}$. For readability, arcs $\{(v_i^j, v_l), (v_l, v_i^j) : 1 \leq i \leq q, 1 \leq j \leq 3, 4 \leq l \leq k\}$ are not drawn.

We claim that there is a solution for our NAE 3SAT instance iff there exists a valid partition $(V_1, V_2, \ldots, V_k)$ of V.

($\Rightarrow$) Given a truth assignment $\mathcal{A}$ of $\mathcal{V}_n$ such that each clause contains at least one true and one false literals, we make a partition $\mathcal{P} = \{V_1, V_2 \ldots V_k\}$ of V as follows: $V_1 = \{v_i^j : c_i^j = \texttt{true} \text{ in } \mathcal{A}\}$, $V_2 = \{v_i^j : 1 \leq i \leq q, 1 \leq j \leq 3\} \setminus V_1$, and, $V_i = \{v_i\}$ for $3 \leq i \leq k$. By definition, there is one or two true literal(s) in each clause. Thus, among the three vertices corresponding to the literals of c_i, at least one and at most two of them are in V_1 (resp. V_2). Moreover, considering $\mathcal{A}$, any variable cannot be simultaneously `true` and `false`. Therefore, two vertices representing complementary literals cannot both belong to V_1 (resp. V_2). Then it follows that each $D[V_i]$, $1 \leq i \leq k$, is a DAG since all cycles have been destroyed. Since G is complete, we have a valid solution.

($\Leftarrow$) Let $\mathcal{P} = \{V_1, V_2 \ldots V_k\}$ be a valid partition of V. We build a truth assignment $\mathcal{A}$ as follows. By construction, there is no loss of generality in assuming

$V_3 = \{v_3\}, \ldots, V_k = \{v_k\}$ (each of these vertices has to occur as a singleton in the partition). Let us define a truth setting $\mathcal{A}$ such that literal c_i^j, $1 \leq i \leq q$ and $1 \leq j \leq 3$, evaluates to `true` iff $v_i^j \in V_1$. This requirement certainly defines a truth setting assignment since opposite literals are part of a cycle of length two. Let us now prove that $\mathcal{A}$ is a solution for our NAE 3SAT instance. Indeed, at least one and at most two of the vertices representing the literals of each clause c_i belong to V_1. Thus, any clause c_i is indeed satisfied and by at most two of its literals. □

Proposition 5. *k-DAGCC-PARTITION is NP-complete even when G is a star.*

Proof. We propose a reduction from the NP-complete k'-INDEPENDENT SET (k'-IS) problem [7]: Given a graph $G_I = (V_I, E_I)$ and a positive integer k', is there an independent set $V_I' \subseteq V_I$ of cardinality at least k' ?

Given any instance G_I of k'-IS, we build D and G as follows:

- $V = V_I \cup \{v_r\}$
- $A = \{(v, v'), (v', v) : (v, v') \in E_I\}$
- $E = \{(v_r, v) : v \in V_I\}$

Moreover, we set $k = |V_I| - k' + 1$. Roughly speaking, graph D is obtained from G_I by replacing each edge by two arcs in opposite directions, and by adding an isolated vertex v_r. The graph G is a star whose center is v_r; thus v_r must be part of any connected subgraph of G of order strictly greater than one. Construction of both these graphs is illustrated in Figure 2.

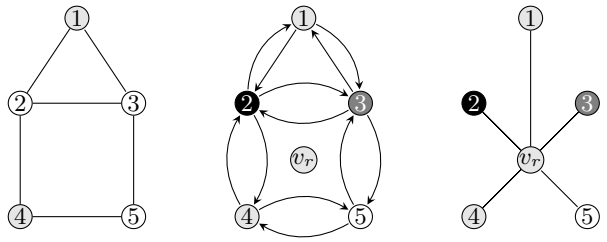

Fig. 2. Illustration of the construction of D and G, given G_I. We highlighted a possible 2-IS $\{1, 4\}$ and a corresponding valid 4-partition of V.

We will show that there is an independent set, in $G_I = (V_I, E_I)$, of cardinality at least k' iff there is a solution for k-DAGCC-PARTITION with $k = |V_I| - k' + 1$.

($\Rightarrow$) Given an independent set of G_I of cardinality greater or equal than k', choose any subset V_I' of k' vertices of it (which is itself an independent set). We compute the partition $\mathcal{P} = \{V_1, V_2 \ldots V_k\}$ of V, where $k = |V_I| - k' + 1$, as follows.

Let $V_1 = V'_I \cup \{v_r\}$. Consider any ordering of the vertices in $V_I \setminus V'_I$, and, for every $2 \leq i \leq k$, let $V_i = \{v\}$, where v is the $(i-1)$-th vertex in this ordering. First, note that any $D[V_i]$, $2 \leq i \leq k$ is indeed a DAG since it is composed of a single vertex. Moreover, by definition, $\nexists \{v, v'\} \subseteq V'_I$, such that $(v, v') \in E_I$. Therefore, $D[V_1]$ is composed of isolated vertices (and thus, is also a DAG). The induced subgraphs $G[V_i]$, $2 \leq i \leq k$, are trivially connected; moreover, connectivity of $G[V_1]$ is ensured by the fact that $v_r \in V_1$.

($\Leftarrow$) Given a solution $\mathcal{P} = \{V_1, V_2 \dots V_k\}$, to k-DAGCC-Partition, we build an independent set $V'_I \subseteq V_I$ of G_I, such that $|V'_I| = |V_I| - k + 1$, as follows. Let $V'_I = V_j \setminus \{v_r\}$ where D_j is the DAG containing v_r. As previously mentioned, in any solution, any connected subgraph of G of order strictly greater than one must contain v_r. Since we require that for every $1 \leq i < j \leq k$, $V_i \cap V_j = \emptyset$, at most one of the V_is can be of order strictly greater than one, and it contains v_r. Thus, $|V'_I| = |V_j| - 1 = ((|V_I| + 1) - (k-1)) - 1$, that is $|V'_I| = |V_I| - k + 1$. Moreover, V'_I is indeed an independent set since D_j is a DAG and thus does not contain cycles. □

Finally, let us prove the inapproximability of Min-DAGCC-Partition when G is a graph or a tree.

Proposition 6. Min-DAGCC-Partition *cannot be approximated within* $n^{1-\epsilon}$, *for any* $\epsilon > 0$.

Proof. We give an L-reduction from the Minimum Chromatic Number (Min-CN) problem defined as follows: Given a graph $G_C = (V_C, E_C)$, find a proper vertex coloring of G_C using the minimum number of colors, where a vertex coloring is said to be proper iff two neighbors in G_C carry different colors.

Given any instance G_C of Min-CN, we build D and G as follows:

- $V = V_C$
- $A = \{(v, v'), (v', v) : (v, v') \in E_C\}$
- $E = \{(v, v') : v, v' \in V_C\} \setminus E_C$

In other words, we keep the same set of vertices as in G_C, the graph D is obtained from G_C by replacing each edge with two arcs in opposite directions, while G is the complement of G_C. An example of such a construction is given in Figure 3.

Let us prove that this construction is indeed an L-reduction from Min-CN. More precisely, we will prove the following property: there exists a proper coloring for G_C using k colors iff there exists a valid cardinality k partition $\mathcal{P} = \{V_1, V_2 \dots V_k\}$ of V.

($\Rightarrow$) Given a coloring of G_C with k colors, let V_i, $1 \leq i \leq k$, be the set of vertices assigned color i. Let us now show that the partition $\mathcal{P} = \{V_i : 1 \leq i \leq k\}$ is valid (i.e., for every $1 \leq i \leq k$, $D[V_i]$ is a DAG and $G[V_i]$ is connected). Indeed, by definition of a proper coloring, each V_i, $1 \leq i \leq k$, is an independent set. Thus, any $D[V_i]$, $1 \leq i \leq k$, is composed of isolated vertices – and therefore is a DAG. Moreover, the induced subgraphs in G are indeed connected since, by construction, whenever two vertices are not adjacent in G_C, they are in G.

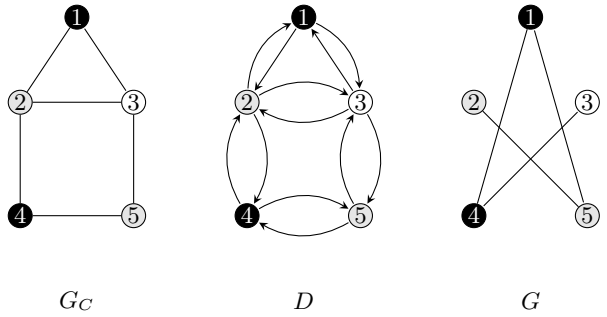

Fig. 3. Illustration of the construction of D and G, given G_C. We highlighted a 3-coloring of D and a corresponding valid 3-partition of V.

($\Leftarrow$) Given a valid k-partition $P = \{V_1, V_2 \dots V_k\}$ of V, we assign to any vertex $v \in V_i$ the color i, for any $1 \leq i \leq k$. This assignment is a proper coloring since, by construction, no $D[V_i]$, $1 \leq i \leq k$, contains an arc, otherwise it would not be a DAG.

The above reduction linearly preserves the approximation, and moreover the sizes of the solutions in the two problems are equal. Hence, given an approximation algorithm for Min-DAGCC-Partition, one can derive an algorithm for Min-CN with the same approximation ratio. Since Min-CN cannot be approximated within $n^{1-\epsilon}$ for any $\epsilon > 0$ [23], so does Min-DAGCC-Partition. □

Proposition 7. Min-DAGCC-Partition *is* APX*-hard even when G is a tree.*

Proof. We give an L-reduction from the APX-hard problem Set Cover-2 [17] defined as follows: Given a ground set $\mathcal{X} = \{x_1, \dots x_n\}$ and a collection of sets $\mathcal{C} = \{S_1, \dots S_q\}$ in which each element of $\mathcal{X}$ appears at most twice, find a minimum set cover $\mathcal{C}'$, i.e. a set $\mathcal{C}' \subseteq \mathcal{C}$ such that $\mathcal{X} = \bigcup_{S_i \in \mathcal{C}'} S_i$ and $|\mathcal{C}'|$ is minimum. In the rest of the proof, for any $1 \leq i \leq q$, we will denote by s_i^j the j-th element of S_i.

Given any instance $(\mathcal{X}, \mathcal{C})$ of Set Cover-2, we build D and G as follows:

- $V = \{v_r\} \cup \{v_i : S_i \in \mathcal{C}\} \cup \{v_i^k : x_k = s_i^j, S_i \in \mathcal{C}, 1 \leq j \leq |S_i|\}$
- $A = \{(v_i^k, v_j^k), (v_j^k, v_i^k) : x_k \in S_i \cap S_j\} \cup \{(v_r, v_i^k), (v_i^k, v_r) : \nexists S_j \text{ s.t. } x_k \in S_i \cap S_j\}$
- $E = \{(v_r, v_i) : S_i \in \mathcal{C}\} \cup \{(v_i, v_i^k) : x_k \in S_i, S_i \in \mathcal{C}\}$.

Otherwise stated, G is a tree rooted at v_r, whose children are the vertices $v_i, 1 \leq i \leq q$ (where v_i, $1 \leq i \leq q$, represents S_i). Each v_i has one child per element of S_i (v_i^k for $x_k \in S_i$). In D, the vertices $v_i, 1 \leq i \leq q$, are isolated whereas the two vertices (v_i^k, v_j^k) representing an element x_k appearing twice in $\mathcal{C}$ form a cycle of length two. Each of the remaining vertices forms a cycle of length two with v_r. An illustration of such a construction is given in Figure 4.

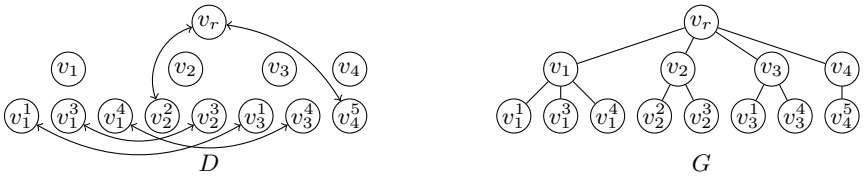

Fig. 4. Illustration of the construction of D and G, given $\mathcal{C} = \{\{x_1, x_3, x_4\}, \{x_2, x_3\}, \{x_1, x_4\}, \{x_5\}\}$.

Let us now prove that this construction is indeed an L-reduction; for this, we show that there is a solution $\mathcal{C}'$ of size k for Set Cover-2 iff there is a valid partition of V in at most $k+1$ sets $V_1, V_2 \dots V_{k+1}$.

($\Rightarrow$) Given a set-cover $\mathcal{C}' \subseteq \mathcal{C}$ of cardinality k, we compute the partition $\mathcal{P} = \{V_1, V_2 \dots, V_k, V_{k+1}\}$ of V as follows: for each $S_i \in \mathcal{C}'$, $1 \leq i \leq k$, let $V_i = \{v_i\} \cup \{v_i^k : x_k \in S_i\}$, and let $V_{k+1} = \{v_r\} \cup \{v_j, v_j^k : x_k \in S_j, S_j \in \mathcal{C} \setminus \mathcal{C}'\}$. In other words, there is a set V_i (inducing the subtree of G rooted at v_i) for each S_i belonging to the set cover; whereas V_{k+1} contains the remaining vertices (including the root v_r of G). Consequently, by construction, for any $1 \leq i \leq k+1$, $G[V_i]$ is indeed connected. Moreover, each $D[V_i]$, $1 \leq i \leq k$, is a DAG since it does not contain v_r and each element of $\mathcal{X}$ occurs at most once in S_i. Finally, $D[V_{k+1}]$ is also a DAG since, by definition of Set Cover-2, any element appears at most twice in $\mathcal{C}$ and at least once in $\mathcal{C}'$. Therefore, in $\mathcal{C} \setminus \mathcal{C}'$, any element appears at most once, and thus no cycle occurs in $D[V_{k+1}]$.

($\Leftarrow$) Given a valid partition $\mathcal{P} = \{V_1, V_2 \dots V_k\}$ of V, we build a cover $\mathcal{C}' = \{S_1, \dots, S_{k-1}\}$ as follows. Assume, wlog, $v_r \in V_1$; then $\mathcal{C}' = \{S_i : \exists v_i^j \notin V_1\}$. In other words, we add in the cover all the sets having an element whose corresponding vertex does not belong to V_1. First, note that, by construction, there is at most one vertex of $\{v_i : 1 \leq i \leq q\}$ in any $G[V_j]$, $2 \leq j \leq k$, since any path linking two such vertices goes through v_r, which is already contained in V_1. It follows that $|\mathcal{C}'| \leq k-1$. It remains to show that $\mathcal{C}'$ is indeed a set cover for $\mathcal{X}$. To do so, consider any element $x_i \in \mathcal{X}$. If x_i occurs exactly once in $\mathcal{C}$ (wlog, suppose $x_i \in S_j$), then the corresponding vertex v_j^i cannot belong to V_1 since, by construction, it would induce a cycle with v_r in $D[V_1]$. Therefore, $S_j \in \mathcal{C}'$. Otherwise, x_i occurs exactly twice, and then at most one of the corresponding vertices v_j^i and $v_{j'}^i$ belongs to V_1, since otherwise it would induce a cycle in $D[V_1]$. Thus, every $x_i \in \mathcal{X}$ appears at least once in $\mathcal{C}'$.

The above reduction is an L-reduction from Set Cover-2 to Min-DAGCC-Partition. Hence, since Set Cover-2 is APX-hard, so is Min-DAGCC-Partition, and the proposition is proved. □

4 Cover Version of k-DAGCC

As we did in the previous section, we now show several complexity results concerning the k-DAGCC-Cover and Min-DAGCC-Cover problems (see also Table 2).

Let us first provide a polynomial-time algorithm when G is a path (Proposition 8). We first prove the following lemma.

Lemma 1. *When G is a path, k-DAGCC-Cover admits a YES answer iff k-DAGCC-Partition admits a YES answer.*

Proof. Any partition being a cover, if k-DAGCC-Partition admits a YES answer, then so does k-DAGCC-Cover. Conversely, consider any positive answer to k-DAGCC-Cover. Hence, there exists a cover $\mathcal{C} = \{V_1, V_2, \dots V_k\}$ of D such that for every $1 \leq i \leq k$, $G[V_i]$ is connected and $D[V_i]$ is a DAG. Since G is a path, so does any $G[V_i]$. Suppose now that a vertex $v \in V$ belongs to at least three pairwise distinct sets V_p, V_q, V_r from $\mathcal{C}$. In that case, G being a path, one of these three sets, say (wlog) V_p, satisfies $V_p \subseteq V_q \cup V_r$, and we can remove V_p from $\mathcal{C}$ and obtain a strictly smaller cover. Applying this rule until no such case occurs, we end up with a cover $\mathcal{C}' = \{V'_1, \dots V'_{k'}\}$, $k' \leq k$, in which every vertex $v \in V$ belongs to at most two different sets of $\mathcal{C}'$. Therefore, assuming the V'_is are ordered according to their leftmost vertex in G, we consider the following partition: $\mathcal{P} = \{V''_i = V'_i \setminus V'_{i+1} : 1 \leq i < k' \text{ s.t. } V'_i \in \mathcal{C}'\} \cup \{V'_{k'}\}$. $\mathcal{P}$ is indeed a partition of V, of cardinality $k' \leq k$; besides, (a) since for any $1 \leq i < k'$, $D[V'_i]$ is a DAG, so does any $D[V''_i]$; (b) $D[V'_{k'}]$ is a DAG by hypothesis, and (c) $G[V''_i]$, $1 \leq i < k'$, and $G[V'_{k'}]$ are all subpaths of G, and therefore connected. Thus k-DAGCC-Partition admits a YES answer, and the lemma is proved. □

Proposition 8. Min-DAGCC-Cover *is polynomial-time solvable when G is a path.*

Proof. By Lemma 1, we know that when G is a path, the (YES/NO) solutions of k-DAGCC-Partition and k-DAGCC-Cover are equivalent. Thus, in the minimization versions of both problems (namely, Min-DAGCC-Partition and Min-DAGCC-Cover), the same minimum value is reached, and in particular if there exists a cardinality k solution to Min-DAGCC-Partition, then there exists a cardinality k solution to Min-DAGCC-Cover. Besides, any partition being a cover, we conclude that any solution to Min-DAGCC-Partition is also a solution to Min-DAGCC-Cover Since Min-DAGCC-Partition is polynomial time solvable (see Proposition 1), so does Min-DAGCC-Cover. □

As opposed to k-DAGCC-Partition, k-DAGCC-Cover proves to be NP-complete even when G is a star and k is a constant.

Proposition 9. *For any $k \geq 3$, k-DAGCC-Cover is NP-complete, even when G is a star.*

Proof. Clearly, k-DAGCC-Cover is in NP. In order to prove that the problem is NP-hard, we provide a reduction from the Minimum Chromatic Number (Min-CN) problem, in its natural decision version: Given a graph $G_C = (V_C, E_C)$ and an integer k, does there exist a proper vertex coloring of G_C using at most k colors ? This problem has been shown to be NP-hard for any $k \geq 3$ [7].

Given any instance G_C of Min-CN, we build D and G as follows:

- $V = V_C \cup \{v_r\}$
- $A = \{(v, v'), (v', v) : (v, v') \in E_C\} \cup \{(v, v_r) : v \in V_C\}$
- $E = \{(v_r, v) : v \in V_C\}$

In other words, D is obtained from G_C by replacing each edge by two arcs in opposite directions, and by adding an arc from any vertex to v_r, while G is a star whose center is v_r.

We now prove the following property: there exists a proper coloring for G_C using k colors iff there exists a valid cardinality k cover $\mathcal{C} = \{V_1, V_2 \dots V_k\}$ of V.

($\Rightarrow$) Given a proper coloring of G_C with k colors, let S_i, $1 \leq i \leq k$, be the set of vertices assigned color k. We compute a cardinality k cover $\mathcal{C} = \{V_1, V_2 \dots V_k\}$ of V as follows: for any $1 \leq i \leq k$, $V_i = S_i \cup \{v_r\}$. By definition of a proper coloring, each S_i is an independent set in G_C (thus in D). Hence, since the out-degree of v_r is equal to zero, any $D[V_i]$, $1 \leq i \leq k$, is a DAG. Moreover, any $G[V_i]$ is indeed connected due to the fact that (i) G is star whose center is v_r, and (ii) every V_i, $1 \leq i \leq k$, contains v_r. Hence, $\mathcal{C}$ is valid.

($\Leftarrow$) Given a valid cardinality k cover $\mathcal{C} = \{V_1, V_2 \dots V_k\}$ of V, we assign to any vertex $v \in V_i \setminus \{v_r\}$ the color i, for any $1 \leq i \leq k$. This assignment is a proper coloring of G_C since, by construction, no V_i can contain two neighbors in G_C, since otherwise, by construction of D, $D[V_i]$ would not be a DAG. □

We finally show that Min-DAGCC-Cover is hard to approximate. Note that this inapproximability even holds when G is a star (instead of general graphs for the problem Min-DAGCC-Partition).

Proposition 10. Min-DAGCC-Cover *cannot be approximated within* $n^{1-\epsilon}$*, for any* $\epsilon > 0$*, even when* G *is a star.*

Proof. The reduction provided in proof of Proposition 9 is actually an L-reduction, since the sizes of the solutions in the two problems are equal (we have indeed proved, following our reduction, that "there exists a proper coloring for G_C using k colors iff there exists a valid cardinality k cover $\mathcal{C} = \{V_1, V_2 \dots V_k\}$ of V"). Hence, given any approximation algorithm for Min-DAGCC-Cover, one can derive an algorithm for Min-CN, with the same approximation ratio. Since Min-CN cannot be approximated within $n^{1-\epsilon}$ for any $\epsilon > 0$ [23], so does Min-DAGCC-Cover.

5 Conclusion

In this paper, we have studied the problem of decomposing (i.e., partitioning or covering) a directed graph D into DAGs, such that each DAG induces a connected subgraph in a given undirected graph G built on same vertex set as D. We have provided, depending on the constraints imposed on the input, several polynomial-time algorithms, as well as hardness and inapproximability results.

There are still several open problems worthwhile to study. For example, one may consider the parameterized complexity of k-DAGCC-Partition, where the parameter is the number of partitions, in the specific case where G is a tree (we indeed know that no FPT algorithm of complexity $f(k)n^{\mathcal{O}(1)}$ is possible when G is a graph, since the problem is NP-complete for $k = 2$). One may also consider studying the approximability of Min-DAGCC-Partition.

References

1. F. Boyer, A. Morgat, L. Labarre, J. Pothier, and A. Viari. Syntons, metabolons and interactons: an exact graph-theoretical approach for exploring neighbourhood between genomic and functional data. *Bioinformatics*, 21(23):4209–4215, 2005.
2. Y.-P. Deniélou, F. Boyer, A. Viari, and M.-F. Sagot. Multiple alignment of biological networks: A flexible approach. In *Proc. 20th Annual Symposium on Combinatorial Pattern Matching (CPM)*, volume 5577 of *Lecture Notes in Computer Science*, pages 263–273, 2009.
3. P. Durek and D. Walther. The integrated analysis of metabolic and protein interaction networks reveals novel molecular organizing principles. *BMC Syst Biol*, 2(1), 2008.
4. G. Fertin, H. Mohamed Babou, and I. Rusu. A pattern-guided approach to compare heterogeneous networks. Available at `http://pagesperso.lina.univ-nantes.fr/~E09D478T/SGM-DB.pdf`. Submitted, 2011.
5. J. Flannick, A. Novak, B. S. Srinivasan, H. H. McAdams, and S. Batzoglou. Graemlin: General and robust alignment of multiple large interaction networks. *Genome Res*, 16(9):1169–1181, 2006.
6. I. Gamzu, D. Segev, and R. Sharan. Improved orientations of physical networks. In *Proc. 10th Workshop on Algorithms in Bioinformatics (WABI 2010)*, volume 6293 of *Lecture Notes in Bioinformatics*, pages 215–225, 2010.
7. M. Garey and D. Johnson. *Computers and Intractability: A guide to the theory of NP-completeness*. W.H. Freeman, San Francisco, 1979.
8. A. Gitter, J. Klein-Seetharaman, A. Gupta, and Z. Bar-Joseph. Discovering pathways by orienting edges in protein interaction networks. *Nucleic Acids Research*, 39(4):e22, 2011.
9. C. Huthmacher, C. Gille, and H. Holzhütter. A computational analysis of protein interactions in metabolic networks reveals novel enzyme pairs potentially involved in metabolic channeling. *J Theor Biol*, 252(3):456 – 464, 2008.
10. B. P. Kelley, R. Sharan, and R. M. Karp *et al.* Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proc Natl Acad Sci USA*, 100(20):11394–11399, 2003.

11. B. P. Kelley, B. Yuan, F. Lewitter, R. Sharan, B. R. Stockwell, and T. Ideker. Pathblast: a tool for alignment of protein interaction networks. *Nucleic Acids Res*, 32(Web Server issue), 2004.
12. O. Kuchaiev, T. Milenkovic, V. Memisevic, W. Hayes, and N. Przulj. Topological network alignment uncovers biological function and phylogeny. *J R Soc Interface*, 7(50):1341–1354, 2010.
13. I. Lee, S. V. Date, A. T. Adai, and E. M. Marcotte. A probabilistic functional network of yeast genes. *Science*, 306:1555–1558, 2004.
14. A. Medvedovsky, V. Bafna, U. Zwick, and R. Sharan. An algorithm for orienting graphs based on cause-effect pairs and its applications to orienting protein networks. In *Proc. 8th Workshop on Algorithms in Bioinformatics (WABI 2008)*, volume 5251 of *Lecture Notes in Bioinformatics*, pages 222–232, 2008.
15. M. Narayanan and R. M. Karp. Comparing protein interaction networks via a graph match-and-split algorithm. *J of Comput Biol*, 14(7):892–907, 2007.
16. C. Pal and L. Hurst. Evidence against the selfish operon theory. *Trends Genet*, 20:232–234, 2004.
17. C. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *J Comput Syst Sci*, 43(3):425–440, 1991.
18. R. Y. Pinter, O. Rokhlenko, E. Yeger-Lotem, and M. Ziv-Ukelson. Alignment of metabolic pathways. *Bioinformatics*, 21(16):3401–3408, 2005.
19. S. Rison, S. Teichmann, and J. Thornton. Homology, pathway distance and chromosomal localisation of the small molecule metabolism enzymes in Escherichia coli. *J Mol Biol*, 318:911–932, 2002.
20. R. Sharan and T. Ideker. Modeling cellular machinery through biological network comparison. *Nature Biotechnol*, 4(4):427–433, 2006.
21. W. Tian and N. F. Samatova. Pairwise alignment of interaction networks by fast identification of maximal conserved patterns. In *Proc. 14th Pacific Symposium on Biocomputing (PSB)*, pages 99–110, 2009.
22. Y. Zheng, J. Szustakowski, L. Fortnow, R. Roberts, and S. Kasif. Computational identification of operons in microbial genomes. *Genome Res*, 12:1221–1230, 2002.
23. D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(1):103–128, 2007.